GENERATION OF AN OBJECT PROCESSING PLATFORM BETWEEN TWO COMPUTERS BY JOINING SCREENS

Description

Generation of an object processing platform between two computers by assembling screens

The invention relates to a method for generating an object processing platform between an object computer and a processing computer, wherein an ad hoc assembly of screens is performed by the object computer with the processing computer in order to couple their input and/or output means.

Currently, a file transfer between two terminal devices, for example between PDAs (PDA: Personal Digital Assistant), between PCs (PC: Personal Computer) or between PDA and PC, requires a certain amount of overhead. Computers are to be understood generally as also meaning mobile terminals of communications technology, such as, for example, cell phones or mobile telephones.

A great deal of software must be installed in the computers in order to perform a file transfer of this kind or a general object transfer. After this it is only possible to perform the file transfer between these two terminal devices. At the same time the user must understand the technology via which he or she wishes to perform the transfer. This means the user has to start a Bluetooth manager if he or she wants to transfer the file via Bluetooth. Specifically, he or she must select the file to be transferred in this manager and determine the destination. It may also be necessary to select a specific

conversion format. An analogous procedure applies to cable-connected and infrared transmission.

On the other hand, techniques and methods are known for connecting two or more screen to form a large-area display as well as for coupling the input means, such as, for example, mouse and keyboard.

There are many different reasons for using a spontaneous screen assembly or a spontaneous combination or joining together of display devices (ad hoc collaboration display). A screen assembly is to be understood as meaning, for example, the combining of a plurality of screens to create what is referred to as a large-area display. Furthermore the linking of displays, as a standalone device or integrated in a data processing system, is also to be generally understood as such.

The following statements concentrate on the graphical control of screens or displays, basically involving the drawing of objects on a display. Apart from said control of screens, the techniques used for this also include the control of input means, such as, for example, keyboards, mice and the like. All these means for a user interface serving for interaction for an electronic data processing device as well as for a stationary or mobile communication terminal are integrated in the same processing layer in virtually all operating systems.

The starting point is for example the display of a mobile telephone or of a PDA (Personal Digital Assistant).

Thus, for example, a group of people, referred to as an ad hoc community, wish to view a document together in a collective environment or even work collaboratively in said document. A

document, in this context, is to be understood as meaning any representation of a file.

Other persons with, for example, mobile phones wish to process their data on a shared large-area display, skipping the synchronization step.

In a domestic setting the occupants want to be able to view all the contents or statuses of the devices and appliances contained in the home on a central display. These contents include, for example, incoming SMS messages to the cordless or corded telephones or the messages from a running washing machine or dishwasher which are displayed on the screen of a television set.

Furthermore, devices without their own display can be controlled via a mobile phone's display brought along, as it were, by the user.

The following techniques are known for controlling display devices or displays.

The typical method of operation of an operating system OS for controlling a screen SCR or display device is described with reference to Figure 1.

A computer or PDA normally has a single display. The operating system OS accesses an object library WSL (WidgetSet Library). In addition to the operating system OS, as the standard application, as it were, the applications APP installed in the computer generally make use of the object library WSL. Based on the addressing carried out by the applications APP, the object library WSL generates the desired objects, which is to say it

draws the objects, and passes these on to the screen driver SDD (Screen Device Driver). In addition to icons and other symbols, graphic characters and other displayable characters should also be understood to mean objects or interaction objects.

The screen driver SDD edits the objects for the graphics card GC, which then controls the screen SCR directly and displays the objects thereon.

It is not possible to change the screen size.

Figure 2 shows a modern operating system OS, such as, for example, Win2000, WinXP, Linux-X11R6-Xfree86 and others. An operating system OS of this kind can control multiple screens SCR (known as Xinerama feature in Xfree86). For this purpose the operating systems OS generally use a virtual layer, known as a virtual screen driver VSDD. Said virtual screen driver VSDD is inserted between the object library WSL and the screen driver or screen drivers SDD. The virtual screen driver VSDD operates as a front-end connecting element for two, as shown in the figure, or more screen drivers SDD for the simultaneous control of the same number of displays SCR.

The virtual screen driver VSDD imitates or, more precisely, simulates a single screen SCR for the object library WSL, although said single screen has twice the height or twice the width of a single one of these screens SCR. Height or width are in this case dependent on the settings selected by the user. The same applies analogously to more than two screens SCR.

The virtual screen driver VSDD handles the task of passing the objects output by the object library WSL via the associated screen driver SDD to the corresponding graphics card SC, and

thus enables the objects to be displayed at the correct position on one of the screens SCR. The representation is completely transparent to the application APP and can be moved freely over the two screens SCR and stretched and extended over both screens SCR. The virtual screen driver VSDD handles the two screens SCR as a single, physically present screen SCR with twice the size.

In this case only individual screens SCR can be joined together to form a large-area image, said screens being controlled from one and the same platform.

Figure 3 shows a variant in which the computer and the display device SCR are no longer at the same location, but are connected to each other via what is known as a client/server application. An X client XC, for example an X11R6 client, is disposed in the local computer and receives the corresponding data records from the applications APP or, as the case may be, the operating system OS.

The computer, more particularly the X client XC, is connected to an X server XS via a network NL (Network Layer). The network layer NL can be implemented by means of a wired or wireless communication network or a computer connection network. The data communication via the network layer NL takes place on the basis of a correspondingly embodied protocol which does not need to be dealt with in further detail here. This configuration largely corresponds to the configuration shown in Figure 1, with the client/server application, consisting of the X client XC, the network layer NL and the X server XS, being disposed between the applications APP, the operating system OS and the following object library WSL.

The object library WSL with the front-end X server XS, the screen driver SDD or, as the case may be, the virtual screen driver VSDD, and the graphics card SC as well as the screen SCR are embodied as a further computer or as a remotely located computing system.

In a variant hereto the screen driver SDD is replaced by the virtual screen driver VSDD already described in the foregoing with reference to Figure 2. This enables a plurality of screens SCR to be controlled. (This scenario is not depicted in any greater detail in the figure).

This variant is a hybrid solution in which the local screen controller is replaced by a screen controller controlled via the network. This is, as it were, a remote screen controller.

Figure 4 shows what is referred to as a display controlled over the network. In this arrangement there is, on the one side, a client computer CC having an application APP and an operating system OS, an object library WSL, a screen driver SDD or virtual screen driver VSDD, a graphics card GC and a screen SCR.

Disposed on the other side is a computer CTC (Computer To be Controlled) which is to be controlled or remotely controlled and which has the same units.

The two computers CC and CTC are connected via what is called a virtual network computer (VNC). This network may in principle be the WWW (World Wide Web). The virtual network computer VNC is in the proper sense a protocol which accepts the input and output data, converts it into, for example, a serial form and

sends it to a client application running somewhere in the network.

For the purposes of data exchange or data transfer the client computer CC has a VNC client VNC-C which is connected to a data or communication network via the network layer NL already known from Figure 3. The VNC client VNC-C is integrated into the computer CC analogously to an application APP.

For the purpose of handling the data traffic the computer to be controlled CTC has what is referred to as a VNC spy VNC-S which is likewise connected to the network layer NL. The VNC spy VNC-S is for example directly connected to the (virtual) screen driver (V) SDD in the computer CTC.

This arrangement enables the client application to take full control of the computer to be controlled CTC. In this case the user works on the client computer CC in the same way as if he or she were sitting in front of the screen SCR of the computer to be controlled CTC. When the virtual network computer VNC is used, the data exchange can be interrupted and resumed from a different location, with the display settings, such as number and arrangement of the windows in windows, the position of the mouse pointer, etc. being preserved as they were prior to the interruption.

Microsoft uses a similar configuration to this under the name "pcAnywhere".

With combined screens, objects or files can be moved over the entire virtual screen. The associated application or, as the case may be, the file processing function runs on the control computer.

The object of the invention is to make object transfer between computers more convenient and user-friendly.

This object is achieved according to the invention by the features specified in claim 1.

The invention is described below with reference to an exemplary embodiment depicted in the drawing, in which:

Figure 1 shows a known arrangement for controlling a screen by means of a computer,
Figure 2 shows a known arrangement for controlling a large-area screen composed of individual screens,
Figure 3 shows a known arrangement for controlling a screen remotely from a remote computer,
Figure 4 shows a known arrangement for controlling a screen remotely via a virtual network computer,
Figure 5 shows a possible arrangement for ad hoc assembly of screens,
Figure 6 shows a scenario for combining screens,
Figure 7 shows a further scenario for combining screens, and
Figure 8 shows a scenario for object transfer according to the invention.

The invention is based on - at least - two assembled screens. The method for combining the displays is of secondary significance for the invention.

Figure 5 shows the essential components for a possible screen assembly, a control computer SC with applications APP, an operating system OS and an object library WSL, as described in Figure 1 as a computer or PDA.

The control computer SC has a screen driver client DD-C (Device Driver Client). In a pictorial representation the applications APP make use of the object library WSL in order to draw graphical objects or components. The object library WSL draws these objects onto the screen driver client DD-C.

The screen driver client DD-C passes on the drawn objects, either to a virtual screen driver service in the network NVDD-S (Networked Virtual Device Driver Service) or to a virtual screen driver service in the virtual network VNVDD-S (Virtual Networked Virtual Device Driver Service). Protocols are used between client DD-C and screen driver service (V)NVDD-S for passing on the objects, said protocols in principle tunneling the corresponding data through the layers of the transmission media. The transmission medium is for example the network layer NL.

In the following explanation of the screen combination, the components which are used by the screen driver client DD-C are referred to in abbreviated form for the sake of simplicity.

In a first possible variant the client DD-C uses a virtual screen driver service in the network NVDD-S, which service operates in the network NVDD as what is termed a virtual screen driver. The virtual screen driver in the network NVDD is, together with the screen driver SDD and at least one screen SCR as well as the associated graphics card GC, part of computers UC1 or UC2 (Used Computer). If two screens SCR are used, the already known virtual screen driver VSDD is used.

In a second possible variant the client DD-C uses a virtual screen driver service in the virtual network VNVDD-S, which

service operates in the virtual network VNVDD as what is termed a virtual screen driver. In principle the screen drivers (V)NVDD act in a similar manner to the virtual screen driver VSDD described in Figure 2, which simulates a single screen driver SDD in place of the two physically present drivers SDD toward the object library WSL.

The difference between the two virtual screen drivers NVDD and VNVDD is the following:

- The virtual screen driver in the network NVDD can control the hardware, i.e. the graphics card GC, directly. If the screen driver client DD-C uses the virtual screen driver in the network NVDD, said driver tunnels the information or objects to the driver SDD via the virtual screen driver VSDD as necessary. In this case the screen driver NVDD must run on the computer UC1 or UC2 in which the corresponding graphics card GC is running.

- The virtual screen driver in the virtual network VNVDD searches in the network layer NL for any available virtual screen driver in the network NVDD or for any further virtual screen driver in the virtual network VNVDD. For this purpose the driver VNVDD uses what are referred to as service discovery protocols. If the screen driver client DD-C uses the virtual screen driver in the virtual network VNVDD, said driver takes over control of the assembled display, i.e. of the screen or screens SCR. The screen driver VNVDD can run anywhere in the network layer NL, even on computers which have no graphics card of their own at all.

A scenario for an ad hoc screen assembly is described below. The components used are a PDA (Personal Digital Assistant) and

a personal computer (PC), both of which are networked for example via a WLAN (Wireless LAN). Both systems PDA and PC are equipped for operation with a virtual screen driver in the network NVDD, and the PDA is additionally equipped for operation with the virtual screen driver in the virtual network VNVDD.

The user of the PDA would like, for example, to have at his or her disposal for a certain period of time a larger screen than the PDA can provide him or her. He or she would also like to use the mouse of the PC as input means.

The PDA is started and after a corresponding application is called the graphical initialization routine attempts to establish a connection to a virtual screen driver in the virtual network VNVDD. The screen driver VNVDD searches in the WLAN for virtual screen drivers in the network NVDD and for further virtual screen drivers in the virtual network VNVDD. It will find at least two screen drivers NVDD, the PDA and the PC.

After a corresponding protocol exchange the screen driver VNVDD on the PDA will either propose a configuration for the two screens of PDA and PC, or it will perform a setting in accordance with the user's defaults. In both cases the user will be prompted for approval, for example. Thereafter, the screen combination shown in Figure 6, for example, is assumed. In this case the comparatively small screen 1 of the PDA supplements the display 2 of the PC at the lower left-hand edge.

The screen driver VNVDD detects the object library WSL of the PDA. The object library WSL begins to draw on the screen 1. The user can now decide whether he or she wants to move the

application or the associated characters, such as text, drawing and graphics in general, to the right onto the screen 2. A partial shifting of the object representation is also possible here, so for example one half appears on the display 1 and the other half on the display 2. The user can also use, for example, the mouse and the keyboard of the PC for the inputs, actually on the PDA.

Since the PDA is the device on which the application started by the user is running, all the input/output data is controlled from the PDA alone. At no time does this application run on the PC itself.

In a further scenario (shown in Figure 7) a device is used which has no screen of its own. This device searches, for example periodically or when triggered by an operator at the press of a button, for a virtual screen driver service in the network or for a virtual screen driver service in the virtual network via one or more network interfaces such as Bluetooth, WLAN or Serial, etc. As soon as a virtual screen driver service is found in the network, the main application begins drawing the objects. The screen driver service runs for example on the notebook or the PDA of a further network user. In this case the protocol is the same as that previously used, with the sole difference that the further user can control the device remotely, being able to switch it off, for example.

The combined screen is shown in Figure 7, with the display 1 of the device, projected, as it were, over the network, being located within the larger (in terms of surface area) display 2 of the notebook.

The screen assembly enables GUI (Graphical User Interface) objects to be moved over or positioned on the area of the two displays as though a single, enlarged display were present. In addition, the input means of the terminal devices involved are at the user's disposal. The technology used for the implementation is of secondary significance for the invention.

According to the invention, an interaction area IA1 and IA2, respectively, is inserted on the displays 1 and 2 involved in the screen assembly. Said area IA1, 2 is, for example, a bar at the top edge of each screen, as shown in Figure 8. The interaction areas IA1 and 2 can also be implemented embodied in terms of area or by means of fields on a touch-sensitive screen; in principle a key combination is also feasible. The operating principle of the interaction areas IA1, 2 is described in more detail below.

If the user moves a positioning symbol, for example a mouse pointer, into the interaction area IA1 or IA2, the associated display 1 (for PDA) or 2 (for PC) changes into the usual local PDA or PC display. This display 1 or 2 then no longer shows the corresponding part of a combined or virtual display.

If the assembled screen has the configuration shown in Figure 7, no additionally designated interaction area needs to be present. In this case the combined screen is a window on the display 2. It is then possible to position the mouse pointer outside of the display 1, i.e. on the desktop of the associated PC for example.

In the following scenario (see Figure 8) the user transfers an object from the PDA (belongs to the display 1) to the PC

(belongs to the display 2). An object of this kind may be a file, an image or, for example, also a clipboard.

In a first step, using one of the above described techniques, the user generates an assembled display from the device-specific displays 1 and 2.

In the next step the user moves an object, for example a document, from screen 1 across to the interaction area IA2 of screen 2, for which purpose he or she uses the DragNDrop technique familiar from Windows.

After a typical waiting period or as a result of the object being placed on the interaction area IA2, the display 2 switches to the customary local display mode for the associated terminal device, in this case for the PC. Finally, a file processing platform is generated by the local coupling of object and interaction area IA2.

In this case there is no file transfer from PDA to the PC involved; instead, the original file remains on the PDA and only the processing is performed on the PC, with the terminated file being stored back on the PDA once more.

In a GUI-oriented system the selected application can be started by placing the transferred object on an associated icon.

In this way a logical link is established between an application running on the PC and the object or the associated data on the PDA. If necessary the moved file is also converted. Thus, for example, a typical wrd document under Psion (for PDA)

will be converted into a typical doc document under Windows (for PC).

Any desired object bus technology, such as, for example, SOAP, JINI, etc., can take over the object and call, process and save back the associated data.